Assignment no: - 5

Problem Statement: Develop a scene in Unity that includes a sphere and plane. Apply rigid body component, material and Box collider to the game Objects. Write a C# program to grab and throw the sphere using VR controller.

To develop a scene in Unity that includes a sphere and plane, apply the Rigid Body component, material, and BoxCollider to the game objects, and implement a C# program to grab and throw the sphere using a VR controller, you can follow these steps:

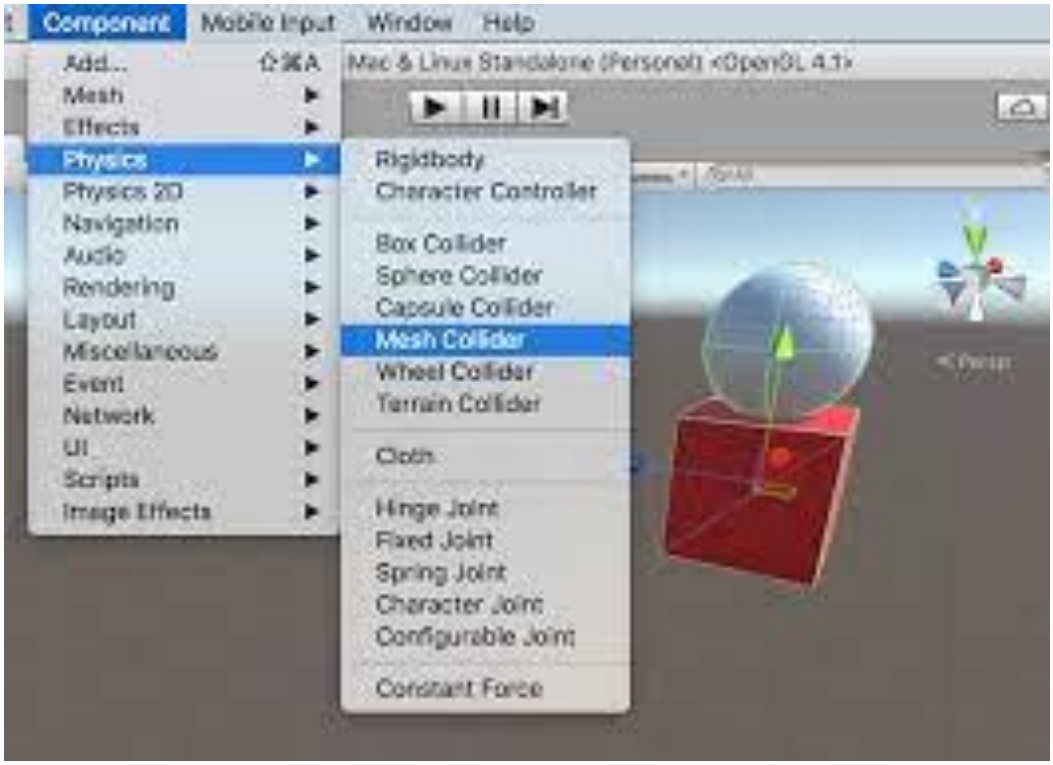
Step 1: Set up the Unity scene

- Create a new Unity project and open it in the Unity Editor.

- Create a new scene by selecting "File" > "New Scene" from the menu bar.

- In the Hierarchy panel, right-click and select "3D Object" > "Sphere" to create a sphere GameObject.

- Similarly, create a plane Game Object using the same method.



Step 2: Apply Rigid Body component, material, and BoxCollider

- Select the sphere Game Object in the Hierarchy panel.

- In the Inspector panel, click on "Add Component" and search for "Rigid Body".

- Add the Rigid Body component to the sphere.

- Adjust the Rigid Body properties as desired, such as mass, drag, and angular drag.

- Next, click on "Add Component" again and search for "Box Collider".

- Add the Box Collider component to the sphere.

- Adjust the Box Collider properties as desired, such as size and center.

- To apply a material to the sphere, select the sphere Game Object in the Hierarchy panel.

- In the Inspector panel, click on the small circle next to the "Material" property.

- Choose or create a material and assign it to the sphere.

- Similarly, you can apply a material to the plane Game Object by selecting it and assigning a material in the same way.

Step 3: Set up VR controller input

- Assuming you have a VR controller set up, you'll need to configure the input for grabbing and throwing the sphere.

- In the Unity Editor, go to the "Edit" > "Project Settings" > "Input" menu.

- Create new Input Axes for "Grab" and "Throw" by clicking on the "+" button.

- Configure the Input Axes with the desired buttons or triggers on your VR controller.

Step 4: Write a C# script for grabbing and throwing

- In Visual Studio or any other C# editor, create a new C# script by selecting "File" > "New" > "File" (or "Class") from the menu bar.

- Rename the script to something like "SphereGrabber" and open it.

- Inside the script, write the following code:

CODE:-

```
using UnityEngine;


public class SphereGrabber : MonoBehaviour
{
    private bool isGrabbing = false; // Flag to indicate if the sphere is currently grabbed
    private Rigid Body grabbedRigid Body; // Reference to the grabbed sphere's Rigid Body component


    private void Update()
    {
        if (Input.GetButtonDown("Grab"))
        {
            if (!isGrabbing)
```

```
        {
            RaycastHit hit;
            if (Physics.Raycast(transform.position, transform.forward, out hit))
            {
                if (hit.collider.CompareTag("Sphere"))
                {
                    grabbedRigid Body = hit.collider.GetComponent<Rigid Body>();
                    isGrabbing = true;
                }
            }
        }
        else if (Input.GetButtonUp("Grab"))
        {
            if (isGrabbing)
            {
                grabbedRigid Body.velocity = transform.forward * 10f; // Adjust the throwing force as desired
                grabbedRigid Body = null;
                isGrabbing = false;
            }
        }
    }
}
```

```

Step 5: Attach the script and tag the sphere

- In the Unity Editor, go to the "Project" panel and locate the script you created ("SphereGrabber.cs").

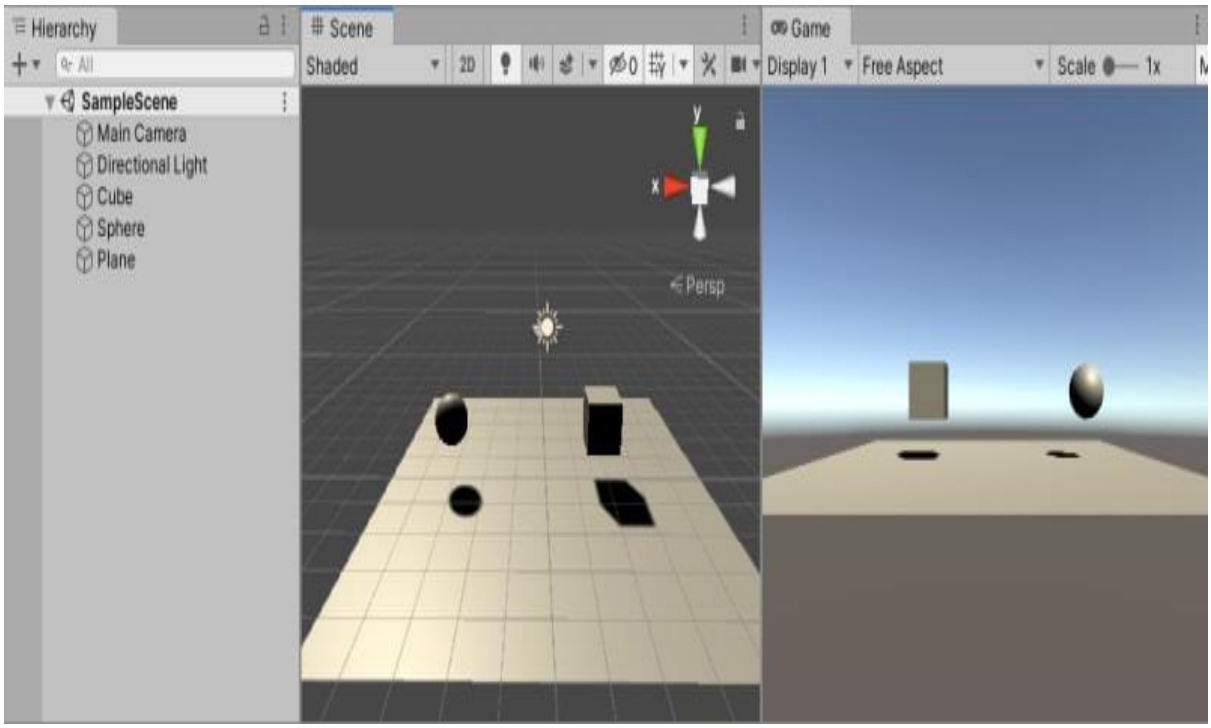
- Drag and drop the script onto the VR controller Game Object to add it as a component.

- Tag the sphere Game Object as "Sphere" by selecting it and going to the Inspector panel. In the "Tag" dropdown, select "Add Tag" and create a new tag named "Sphere". Assign the "Sphere" tag to the sphere GameObject.

Step 6: Testing

- Save your scene and return to Unity.

- Press the Play button in the Unity Editor to run the scene.


```